NASA TECHNICAL NOTE

NASA TN D-7607

NASA TN D-7607

DIGITAL IMAGE REGISTRATION METHOD BASED UPON BINARY BOUNDARY MAPS

by R. R. Jayroe, J. F. Andrus, and C. W. Campbell

George C. Marshall Space Flight Center
Marshall Space Flight Center, Ala. 35812

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • MARCH 1974

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

FOREWORD

This study was initiated under the direction of Dr. R.R. Jayroe, Flight Data Statistics Office, Aerospace Environment Division, Aero-Astrodynamics Laboratory, Marshall Space Flight Center. This work was accomplished via the in-house efforts of Dr. R.R. Jayroe and Mr. C.W. Campbell of the same office, and by the efforts of Dr. J.F. Andrus of Northrop Services, Inc., Huntsville, Alabama, under contract NAS8-21810. Dr. Andrus must be credited for conceiving the original idea used in the binary correlation routine, and has continued to contribute to this study since joining the faculty of Louisiana State University, New Orleans, in August 1973.

DIGITAL IMAGE REGISTRATION METHODS BASED UPON BINARY BOUNDARY MAPS

I. INTRODUCTION

The ability to register or match the ground scene of images in the form of digital data serves an important role in the analysis of remotely sensed, multispectral, earth observation data and change detection. In the case of multispectral camera data, the images from the different camera stations must be registered before a proper analysis can be performed. Registration is also necessary when the data are acquired from several different sensors. In the case of change detection, where data are acquired from the same ground scene but at different time intervals, registration has an important role in determining what changes in shape have taken place in the ground scene, and registration permits an analysis to determine how the multispectral signatures of various ground features change as a function of time. For meteorological applications, registration methods based on meteorological satellite imagery can be used to estimate cloud velocities for global weather information.

Because of the typically large volumes of data involved, the use of binary maps in registration is recommended for the reasons discussed below. Converting the raw data to a binary boundary map typically represents a data compression of 60 to 70 percent of the original data. An additional significant compression of data is realized by working with sequences of, rather than individual, boundary points, which requires only the knowledge of the start scan and column and the length of the sequence.

If $A_1(x,y)$ and $A_2(x+\xi, y+\zeta)$ represent the amplitudes of the data in images 1 and 2 respectively, at scan coordinates x and $x+\xi$ respectively, and at column coordinates y and $y+\zeta$ respectively, then the average product $\overline{A_1(x,y)\,A_2(x+\xi, y+\zeta)}$ is computed over x and y for various combinations of ξ and ζ to determine the scan and column shifts necessary to register the two digital images. If this average is applied to the raw data, it may also be necessary to remove mean values and normalize in order to produce a correlation coefficient with a well defined peak. The use of the raw data appears to have some drawbacks as compared to using binary border maps. First, an ambiguity can result, since a large negative correlation peak can occur and represent a good match as well as a large positive peak. This is in most part due to the variation of the data acquired under different environmental conditions; i.e., different sun angles, seasons, atmospheric conditions, etc. On the other hand, binary boundary maps are produced from relative changes in the data, and the boundaries indicated in the ground scene tend not to change with environmental conditions. Since the binary boundary maps contain data that consist only of 0's and 1's, the average product $\overline{A_1(x,y)\,A_2(x+\xi, y+\zeta)}$ will always be positive and tends to be sharp so that removal of means and normalization is not necessary. In addition, it is possible to compute the average product $\overline{A_1(x,y)\,A_2(x+\xi, y+\zeta)}$ without multiplying when binary data are used. Addition can be used to replace multiplication, and this reduces computer computation time significantly.

Section II is a discussion of the production of binary boundary maps, while Sections III and IV are concerned with the correlation and registration schemes, respectively. Section V contains the results, test cases, and summary

II. BINARY BOUNDARY MAPS

The purpose of the binary boundary map is to categorize the digital data representing the ground scene into homogeneous areas and boundaries, and an example of a boundary map from ERTS digital imagery with some feature identification is shown in Figure 1.

The computer program used to generate the boundary map typically runs about 13 minutes (IBM-7094 time) on an area of 1000 scans by 255 columns, which is approximately 327 samples/second. This time includes the amount of time required to read and compile the program in addition to the calculation time. At present, no concerted effort has been made to optimize the running time of this program or to specialize its output for the registration program, but the efforts are in a holding status for future consideration.

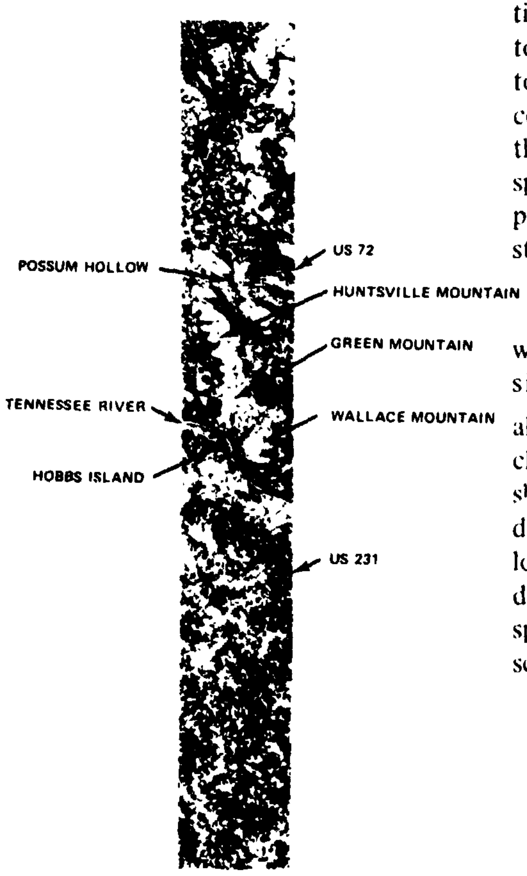

Figure 1. Boundary map from a portion of ERTS imagery.

The logic of the program is to work with two scans of raw digital data simultaneously. Let ${}_kx_{ij}$ represent the algebraic value of the data in the kth channel of data at scan i and column j as shown in Figure 2. If there are n channels of data, then k ranges from 1 to n. For each location i, j, the average vector component distances, which represent various degrees of spectroscopic changes within the ground scene, are computed using the formulas

$$S_x = \left[\frac{1}{n} \sum_{k=1}^{n} ({}_kx_{i,j} - {}_kx_{i-1,j})^2 \right]^{1/2} \tag{1}$$

and

$$S_y = \left[\frac{1}{n} \sum_{k=1}^{n} ({}_k x_{i,j} - {}_k x_{i,j-1})^2 \right]^{1/2} \quad , \qquad (1)$$

(Concluded)

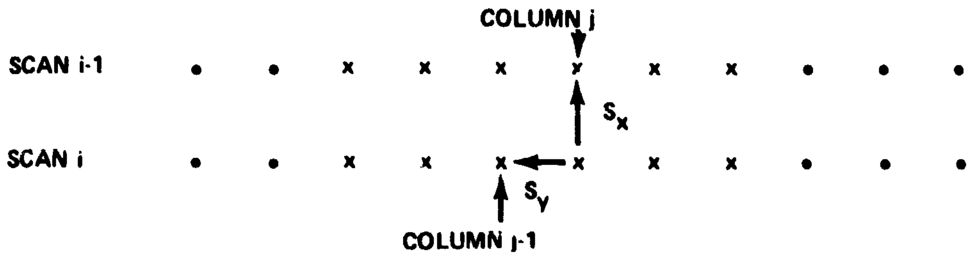

Figure 2. Computation of S_x and S_y for channel K.

and are stored in a joint histogram, $P(S_x, S_y)$. Dividing by n in equation (1) assures that S_x and S_y have the same typical range regardless of the number of data channels. Based upon previous experience with various types of data, it was found that the size of the joint histogram array could be limited in size to a 50 by 50 array such that any data element with a value of S_x or S_y larger than 50 could be automatically assigned as a boundary element. As the raw data are read into the program and the number of occurrences of S_x and S_y are accumulated in the joint histogram, a new tape is created for later use. This tape contains the numbers 1 through $50^2 = 2500$ and instead of writing S_x and S_y on this tape for each data element, one number is written on the tape that gives the location of S_x and S_y in the joint histogram. The location, I, of S_x and S_y in the joint histogram is computed using

$$I = (51) S_x + S_y \quad , \qquad (2)$$

and is a unique one-dimensional representation of the two dimensions S_x and S_y. The new tape containing a number I for each data element eliminates the necessity for recalculating S_x and S_y for each data element a second time. After all the data elements from the data set have been exhausted and the joint histogram is complete, a decision is made as to which combinations of S_x and S_y are to be considered as boundaries. The decision curve for a boundary element is based upon the formula

$$\left(\frac{S_x}{S_x' + ASCN}\right)^{IPOW} + \left(\frac{S_y}{S_y' + ACOL}\right)^{IPOW} > (2)BLIM \quad , \qquad (3)$$

where S_x' and S_y' are the values of S_x and S_y at the mode of the joint histogram and ASCN, ACOL, IPOW, and BLIM are input parameters to the program. The input parameters allow for a wide variety of decision curve shapes and positions. Nominally IPOW = 2 and BLIM = 1, whereas ASCN is usually equal to ACOL and must be estimated based upon experience. The possible values of I or correspondingly S_x and S_y are then inserted into equation (2) and the variable N(I) in the computer program is set equal to 0 if equation (2) is not satisfied and is set equal to 1 if equation (2) is satisfied. The tape containing I for each data element is then read into the program, and the value of N(I) is written out as another tape which is called the binary boundary map and contains only the numbers 0 and 1. The use of the intermediate tape containing the I values permits the boundary decision curve to be varied without recalculating S_x and S_y for every data element.

III. CORRELATION SCHEME

The development work for the correlation scheme was initiated because of a lack of a readily available registration program and after a review and discussion of some of the most recently published papers on registration, such as References 1 through 3. A computer listing of the correlation scheme is provided in Appendix A. The central idea is to compute only those correlation delays which are affected by overlapping boundary elements for each shift of the window against the picture.

Before describing the correlation scheme, it is necessary to define the computer program variables and terminology for the computer routine.

Picture	A section of a boundary map to be used as the reference data.
Window	A section of a boundary map which is to be registered with the picture. The window has one-half as many scans and columns of data as the picture.
NROW	The number of data scans in the window.
NCOL	The number of data samples per scan or columns of data in the window. Nominally, NROW is equal to NCOL.

$\frac{\text{NCOR (I,J)}}{\text{(NROW)(NCOL)}}$	The average product $\overline{A_1(x,y)\ A_2(x+I,\ y+J)}$. The value of NCOR(I,J) is the number of coinciding boundary points that occur when the upper left corner element of the window is placed upon the Ith row and Jth column of the picture. (I = 1,2, ..., NROW + 1 and J = 1, 2, ..., NCOL + 1.)
KMAX	Total number of boundary sequences in the window.
LMAX1	Total number of boundary sequences in the picture for I = 1, 2, ..., NROW.
LMAX	Total number of boundary sequences in the picture for all I, I = 1, 2, ..., (2) NROW.
NW(J)	Dummy variable for reading binary boundary map data. (J = 1, ..., NROW for window; J = 1, ..., (2)NROW for picture.)
K	Number of the Kth window boundary sequence (K = 1, ..., KMAX).
L	Number of the Lth picture boundary sequence (L = 1, ..., LMAX).
IW(K)	The value of IW(K) is the scan number minus one of the Kth boundary sequence in the window.
JW(K)	The value of JW(K) is the column number minus one of the beginning of the Kth boundary sequence in the window.
NSQW(K)	The value of NSQW(K) is the number of boundary elements minus one in the Kth boundary column sequence in the window.
IP(L)	The value of IP(L) is the scan number of the Lth boundary sequence in the picture.
JP(L)	The value of JP(L) is the column number of the beginning of the Lth boundary sequence in the picture.
NSQP(L)	The value of NSQP(L) is the number of boundary elements minus one in the Lth boundary column sequence in the picture.
I	The value of I is the scan shift in NCOR(I,J).

JL	The value of JL is the lower value of the column shift J in NCOR(I,J).
JU	The value of JU is the upper value of the column shift J in NCOR(I,J). (J = JL, JL + 1, ..., JU.)
NAD	The value of NAD is the number of coinciding boundary points at a given I and J for the Kth window boundary sequence and the Lth picture boundary sequence.
JM	The value of JM controls the value of NAD. JM causes NAD to decrease as the Kth and Lth boundary sequences decreasingly overlap because of an increase in J from JL to JU.
NU	For a given I, the value of NU is the column number of the end of the Lth boundary sequence in the picture.
NSPAN	For a given I, the value of NSPAN is the column number of the end of the Kth boundary sequence in the window.
MINL	MINL is a dummy variable which changes and represents the smaller of the window or picture boundary sequence minus one.
CALL TIMNOW ( )	Subroutine for calling internal clock to time computation of NCOR(I,J).

The first step of the program is to read in the sections of the binary boundary maps to be considered, skipping all of the nonboundary or zero elements, and storing the start scan, start column, and column length of the sequences in the window and picture. This information is stored consistent with the definition of IW(K), JW(K), NSQW(K), IP(L), JP(L), NSQP(L), KMAX, LMAX1, and LMAX, and results in a considerable compression of the data. The calculation of the above variables is accomplished in the DO 20 nested do-loop for the window and the DO 40 nested do-loop for the picture. The remaining part of the program contains two major calculation segments, the DO 100 nested do-loop and the DO 200 nested do-loop. Because of their similarity it will suffice to explain only one of these do-loops and then point out their differences. The DO 100 segment is concerned with the entire window and the top half of the picture; i.e., I = 1, 2, ..., NROW. The two outer do-loops, DO 100 and DO 80, determine the I and J shifts that cause the individual boundaries of the K window sequences to coincide with the individual boundaries of the L picture sequences. No negative shifts are permitted; i.e., $1 \leqslant I \leqslant NCOL + 1$ and $1 \leqslant J \leqslant NCOL + 1$. Within the DO 100 and DO 80 loops, there are 3 cases to be considered. The first case is the DO 56 loop and it occurs when the minimum length, MINL, of either the window or picture sequence is one boundary element. This occurrence is illustrated in Figure 3 and can result in an addition of at most 1 to NCOR(I,J) for each value of J. The case which probably occurs most often is

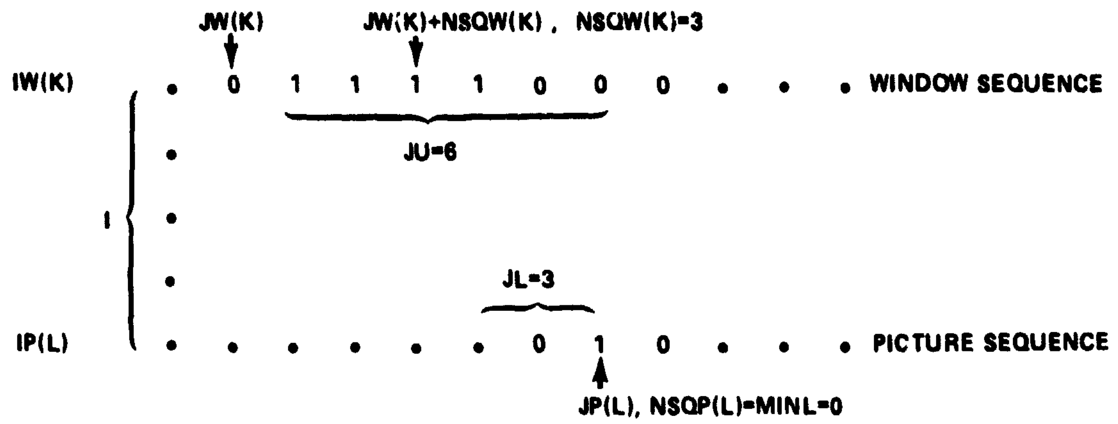

Figure 3. Subcase for DO 56 loop.

the DO 68 loop, and a typical example is shown in Figure 4. The first time through the DO 68 loop NAD = 1 is added to NCOR(I,J) and NAD continues to increase by one until it is greater than MINL, the minimum length of either sequence minus one. Once NAD is one greater than MINL it remains that value until J is greater than JM, and then NAD decreases by one each time until the two sequences no longer overlap in the column direction. Thus, both the initial and final values of NAD added to NCOR(I,J) are one, while the maximum value of NAD is the number of boundary elements in the smaller of the two sequences. Table 1 is a convenient way of viewing the number of occurrences of each column shift due to the presence of both sequences. The top two rows list the possible values of J or the column shift, while the next three rows list the possible values

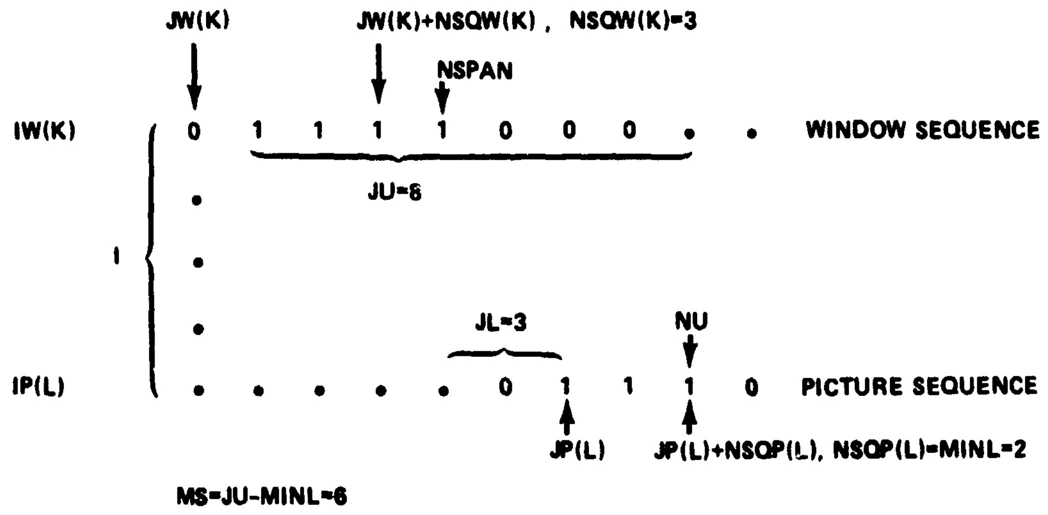

Figure 4. Subcase for DO 68 loop, JL > 0.

TABLE 1. NUMBER OF OCCURRENCES OF EACH COLUMN SHIFT FOR FIGURE 4

	JL			JM		JU
	3	4	5	6	7	8
JP(L)	JL	JL+1	JL+2	JL+3		
JP(L) + 1		JL+1	JL+2	JL+3	JL+4	
JP(L) + NSQP(L)			JL+2	JL+3	JL+4	JL+5
NAD	1	2	3	3	2	1

of J for each boundary element in the picture. The last row is the corresponding value for NAD for each J or the total number of occurrences of each column shift. An additional subcase for the DO 68 loop can occur when JL is negative, JP(L) > JW(K), and the length of the window sequence is equal to or less than the picture sequence. An example of this subcase is shown in Figure 5 which corresponds to Table 2.

The final two subcases occur in connection with the DO 78 loop. Both subcases have the minimum column shift, JL, which is initially negative. The first subcase has JP(L) less than or equal to JW(K) and is illustrated in Figure 6 which corresponds to Table 3. The second subcase is illustrated in Figure 7 which corresponds to Table 4.

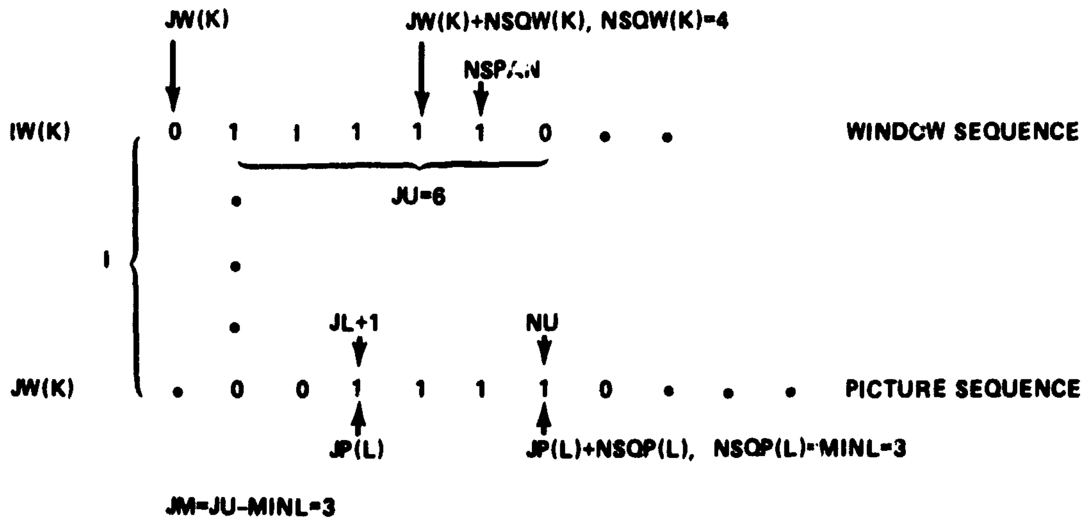

Figure 5. Subcase for DO 68 loop, JL $\leq$ 0, NSPAN $\leq$ NU, JP(L) > JW(K).

TABLE 2. NUMBER OF OCCURRENCES OF EACH COLUMN SHIFT FOR FIGURE 5

	JL		JM			JU
	1	2	3	4	5	6
JP(L)	JL	JL+1	JL+2			
JP(L) + 1	JL	JL+1	JL+2	JL+3		
JP(L) + 2	JL	JL+1	JL+2	JL+3	JL+4	
JP(L) + NSQP(L)		JL+1	JL+2	JL+3	JL+4	JL+5
NAD	3	4	4	3	2	1

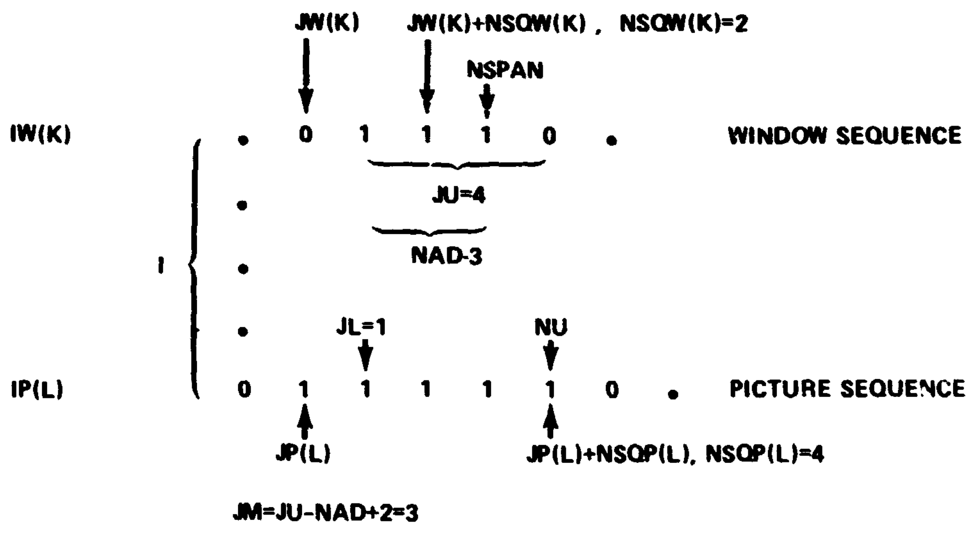

Figure 6. Subcase for DO 78 loop, JL $\leqslant$ 0, JP(L) $\leqslant$ JW(K).

TABLE 3. NUMBER OF OCCURRENCES OF EACH COLUMN SHIFT FOR FIGURE 6

	JL		JM	JU
	1	2	3	4
JP(L) + 1	JL			
JP(L) + 2	JL	JL+1		
JP(L) + 3	JL	JL+1	JL+2	
JP(L) + NSQP(L)		JL+1	JL+2	JL+3
NAD	3	3	2	1

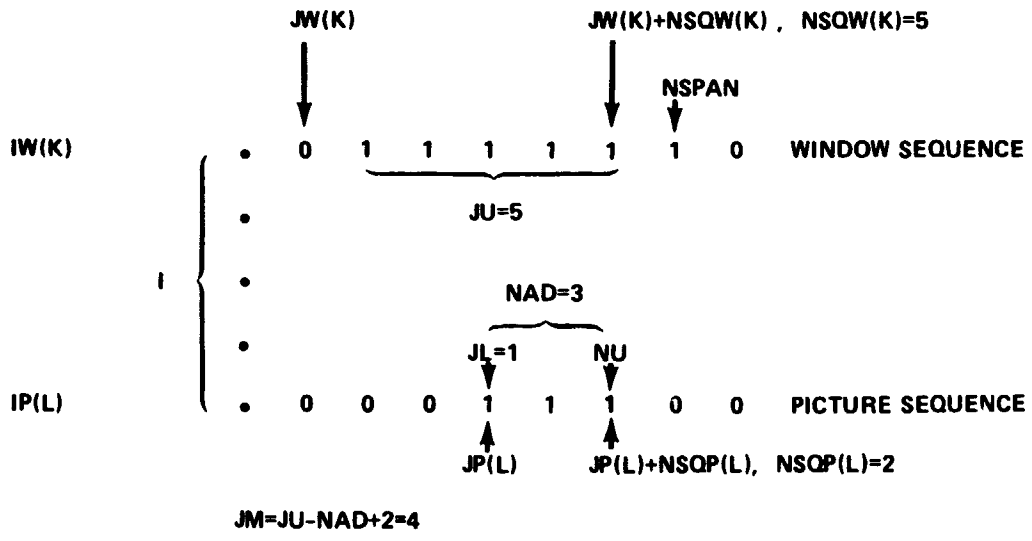

Figure 7. Subcase for DO 78 loop, JL $\leqslant$ 0, JP(L) $>$ JW(K), NSPAN $\geqslant$ NU.

TABLE 4. NUMBER OF OCCURRENCES OF EACH COLUMN SHIFT FOR FIGURE 7

	JL			JM	JU
	1	2	3	4	5
JP(L)	JL	JL+1	JL+2		
JP(L) + 1	JL	JL+1	JL+2	JL+3	
JP(L) + NSQP(L)	JL	JL+1	JL+2	JL+3	JL+4
NAD	3	3	3	2	1

The second segment of the program or the DO 200 loop is identical to the first segment of the program, the DO 100 loop, except for the statement numbers. The main difference is that the DO 200 loop, while considering the entire window, only considers the bottom half of the picture. Thus, the DO 100 loop considers less and less of the first half of the picture sequences for all of the window sequences, and the DO 200 loop considers less and less of the window for the last half of all of the picture sequences. In the DO 100 loop, the scan shift I starts at the maximum available value and decreases to I = 1, while in the DO 200 loop I starts at the minimum available value and increases to a maximum of I = NROW + 1. A scan shift of I = 1 and a column shift of J = 1 indicate that the window and picture are already registered.

IV. REGISTRATION SCHEME

In applying the registration scheme it is assumed that the data can be misaligned by translation and/or rotation, but that no distortion exists between the two digital images. If significant distortions do exist over the entire images, then it may become necessary to register subportions of the digital images as separate data sets.

It is also assumed that both digital images are of the same scale. If the images are not of the same scale, then it is possible to scale down one image by locating several corresponding boundary elements on both images, choose a corresponding reference boundary element on each image, and compute for each image the average distance of all the other boundary elements from the reference boundary element. The ratio of the two average distances gives a scaling factor. The raw data image with the larger average distance can then be scaled to the same size as the other image, and a new boundary map can be computed.

To register two boundary maps, choose one map as the reference map or picture and choose subportions of the other map to be used as windows as illustrated in Figure 8. Figure 9 shows a portion of the two maps overlayed and the minimum and maximum scan and column shifts for the window. For the registration to work properly, the final

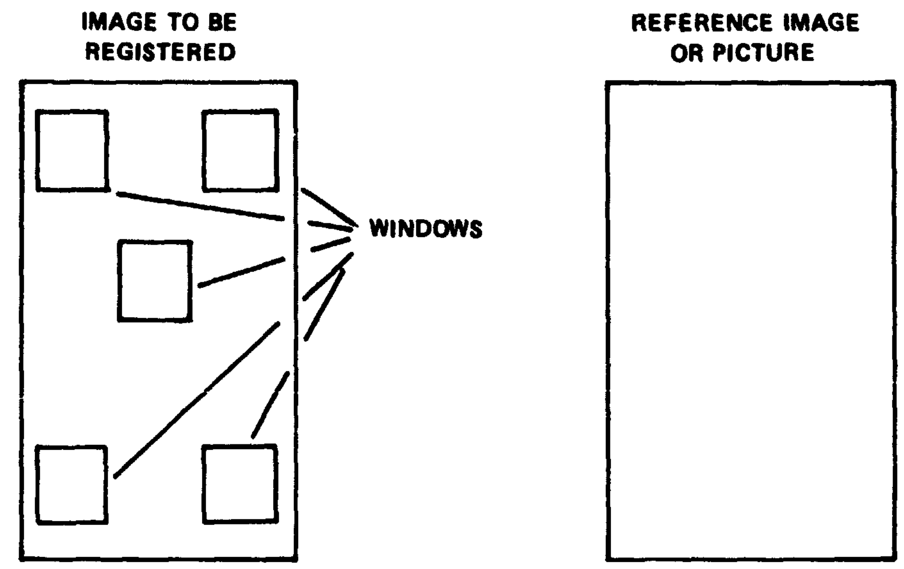

Figure 8. Window selection.

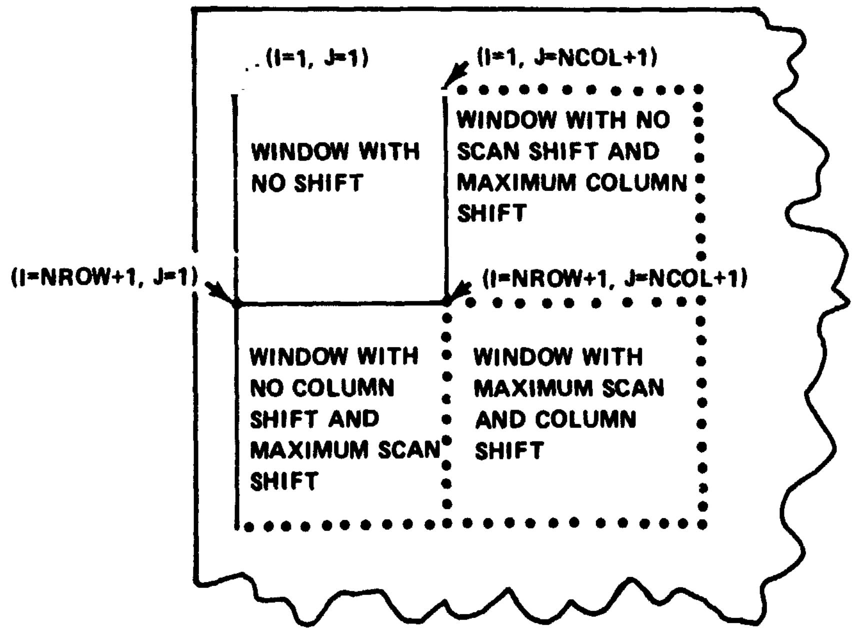
(I=1, J=1)
(I=1, J=NCOL+1)
WINDOW WITH NO SHIFT
WINDOW WITH NO SCAN SHIFT AND MAXIMUM COLUMN SHIFT
(I=NROW+1, J=1)
(I=NROW+1, J=NCOL+1)
WINDOW WITH NO COLUMN SHIFT AND MAXIMUM SCAN SHIFT
WINDOW WITH MAXIMUM SCAN AND COLUMN SHIFT

Figure 9. Overlay of digital images.

two assumptions are that the portions of the picture corresponding to the window are within the allowable range of the window and that the amount of rotation present in the picture or the window can be neglected as far as the computation of NCOR(I,J) is concerned and can be treated as a local translation. If the latter assumption is not valid, it may be necessary to repeat the registration procedure again after the data have been translated and/or rotated based upon the first registration effort. In the event that the first assumption is correct, it should be possible to overcome the second assumption by repeated application of the registration procedure to the data.

The mathematics of the translation-rotation required to register the two images is given below. Let ${}_wx_i$, ${}_wy_i$ be the scan and column coordinates respectively of the upper left corner on the ith window in the image to be registered, and let I_i and J_i correspond to the maximum value of NCOR(I,J) for the ith window. For any one of the windows, for example the jth, let the coordinates

$$ {}_px_j = {}_wx_j + I_j - 1 $$

and (4)

$$ {}_py_j = {}_wy_j + J_j - 1 $$

on the picture be designated as the center of rotation. Thus, if the entire window image is shifted by the amounts I_1-1 and J_1-1, the coordinates $({}_px_j, {}_py_j)$ and $({}_wx_j, {}_wy_j)$ will coincide and the rest of the image can be rotated about these coordinates to complete the registration, provided a rotation exists. If no rotation exists, the images are registered and the maximum value of NCOR(I,J) for the rest of the window occurs at I = J = 1. Assuming that rotation does exist, it will be necessary to compute the following quantities,

$$ \overline{{}_px\,{}_wx} = \frac{1}{N-1} \sum_{i \neq j}^{N} ({}_px_i - {}_px_j)({}_wx_i - {}_wx_j) \quad , \tag{5} $$

$$ \overline{{}_py\,{}_wy} = \frac{1}{N-1} \sum_{i \neq j}^{N} ({}_py_i - {}_py_j)({}_wy_i - {}_wy_j) \quad , \tag{6} $$

$$\overline{{}_px\,{}_wy} = \frac{1}{N-1}\sum_{i\neq j}^{N} ({}_px_i - {}_px_j)({}_wy_i - {}_wy_j) \quad , \tag{7}$$

$$\overline{{}_wx\,{}_py} = \frac{1}{N-1}\sum_{i\neq j}^{N} ({}_wx_i - {}_wx_j)({}_py_i - {}_py_j) \quad , \tag{8}$$

and

$$\tan\theta = -\frac{\overline{{}_px\,{}_wy} - \overline{{}_wx\,{}_py}}{\overline{{}_px\,{}_wx} + \overline{{}_py\,{}_wy}} \quad , \tag{9}$$

where N is the total number of windows. To fetch the data from the image to be registered, it is first necessary to solve for θ, and for each picture coordinate $({}_px, {}_py)$, it is necessary to solve the equations,

$${}_px \cos\theta + {}_py \sin\theta - {}_px_j \cos\theta - {}_py_j \sin\theta + {}_px_j = {}_wx$$

and (10)

$$-{}_px \sin\theta + {}_py \cos\theta - {}_px_j \sin\theta - {}_py_j \cos\theta + {}_py_j = {}_wy$$

for the appropriate unregistered image coordinates. In general, the coordinates $({}_wx, {}_wy)$ will not be integers, but in most cases it is probably most appropriate and less time consuming to round off to the nearest integer coordinate rather than try to interpolate the value of the data between the original data points.

The registration routine is currently being programmed and debugged by MSFC's Computation Laboratory, Engineering Computation Division (S&E-COMP-RRV). The routine has not yet been modified to accept the correlation scheme, but instead accepts a manual input of corresponding boundary elements from two different images.

V. TEST CASES, RESULTS, AND SUMMARY

The binary correlation routine was compared to a correlation and Fast Fourier Transform (FFT) correlation routine by obtaining running times from an internal clock on the IBM-7044 computer. A listing of the correlation routine and a program for using the FFT correlation routine are presented in Appendices B and C. The Fast Fourier Transform used, FFT7, is not presented in this report since it has been evaluated and listed in Reference 4 along with nine other FFT routines. According to Reference 4, FFT7 was the most versatile and ranked third in speed to the fastest routine, FFT3, which was twice as fast but worked only on real data, performed only a (+i) transform, and produced only a half transform. The second fastest transform was FFT8, which was only 25 milliseconds faster than FFT7.

The data set on which the programs were run is shown in Figure 10. In the data set, only the locations of the boundaries are printed out and represented as 1's, and the window and the picture both start in the upper left corner. The size of the data array used for the window is the first NCOL columns by NROW scans, and the picture is twice as large. For the Fast Fourier Transform the window has to be the same size as the picture, and consequently zeros had to be added to the window array. The window and picture data arrays were arranged such that they were already in registration or such that the maximum value of NCOR(I,J) occurred at I = J = 1.

The output of the binary correlation routine for a 32 by 32 window array is shown in Figure 11, and the maximum number of NCOR(I,J) is the total number of boundary elements in the window. Figure 12 is a graph of correlation lag array size versus running time for the binary correlation, Fast Fourier Transform correlation, and correlation routines. The actual running times for each program versus correlation array size are listed in Table 5. Storage requirements for the FFT correlation routine were such that it was not possible to compute 41 by 41 correlation lag points or larger, and the irregular running times as shown in Figure 12 are a result of the FFT routine being more efficient for some array sizes than others. The correlation routine was able to handle 57 by 57 correlation lag points, but the running time was already 5 minutes for 41 by 41 correlation lag points. Since it appears that running times for the correlation routine can easily be extrapolated from Figure 12 and Table 5, no larger size arrays were run.

According to Reference 1, the Sequential Similarity Detection Algorithm (SSDA) mentioned therein is approximately 50 times faster than the FFT correlation method. The factor of 50 is estimated from time ratios relating arithmetic operations on the IBM 360/65 used in computing the numerator of the correlation coefficient. By relating all arithmetic operations to integer adds for a 32 by 32 window and a 256 by 256 picture, the equivalent integer adds for the correlation, FFT correlation, and SSDA respectively were 2.57 X 10^8 and 2.2 X 10^6, which indicate a factor of 100 times faster than the FFT correlation. To compare the binary correlation with the SSDA it was necessary to put a counter in the DO loops containing NCOR(I,J) and to use the formula of

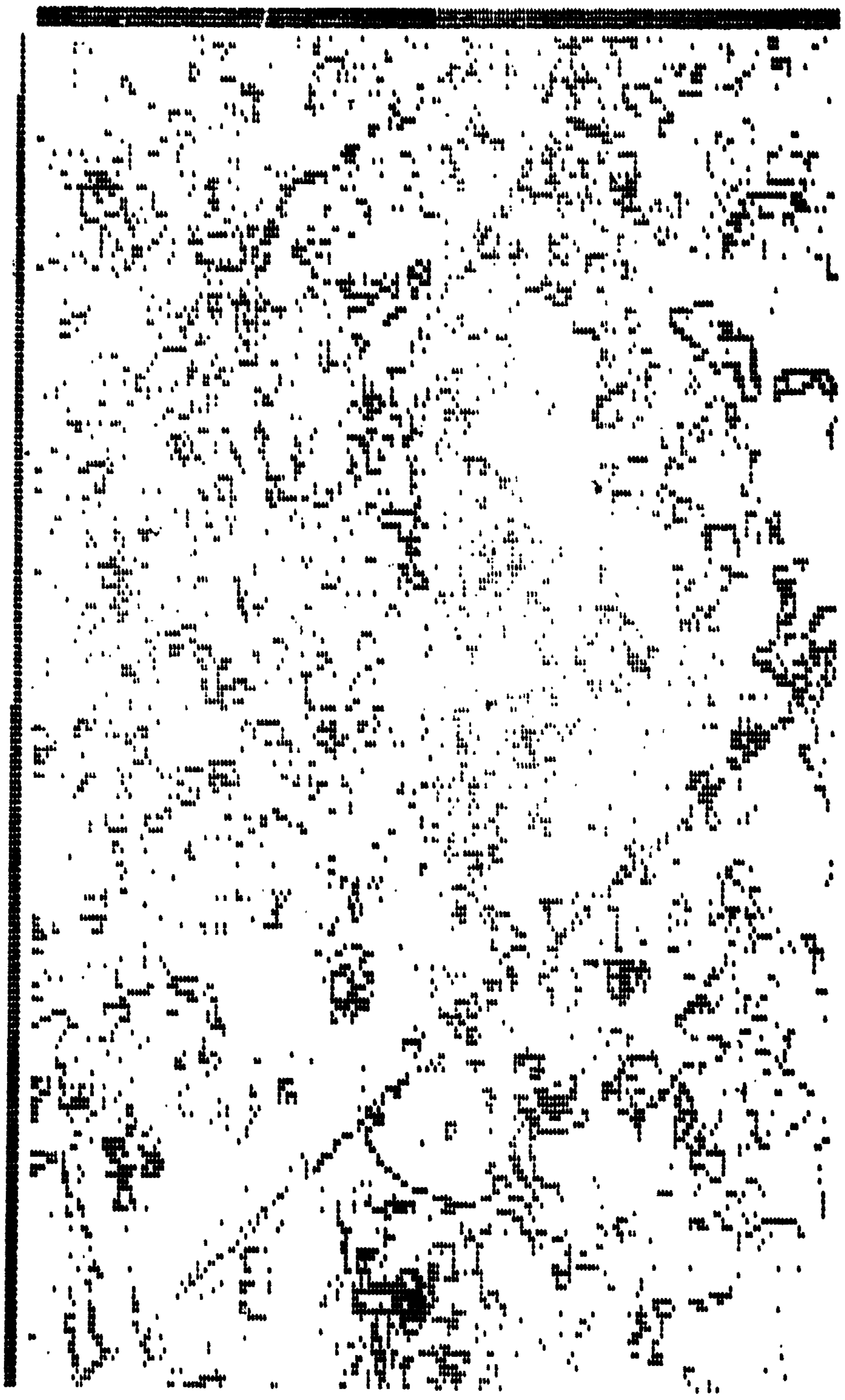

Figure 10. Binary boundary map data set.

```
      CORRELATION MATRIX
  122   54   35   29   33   31   27   30   31   21   23   21   22   21   18   18   12   13   19   18   19   18   17   23   19   21   23   30   24   26   26   29   24
   40   46   34   34   33   29   22   23   26   22   26   26   18   16   16   19   14   12   11    9   15   14   16   18   16   20   23   26   26   25   24   28   27
   33   35   34   33   37   32   27   21   27   21   23   23   24   18   23   19   22   16   23   21   26   24   23   24   25   23   30   32   28   28   33   30   28
   24   28   30   32   30   26   29   25   23   25   19   28   28   26   27   20   23   22   20   23   17   21   23   25   30   29   29   37   38   35   25   33   31
   20   27   33   32   30   31   31   30   33   35   29   26   25   25   23   24   29   29   22   18   18   21   22   22   21   23   25   30   32   33   28   26   27
   22   24   30   23   30   29   30   25   25   32   24   21   28   28   28   26   23   24   22   21   21   19   22   16   17   23   17   25   24   29   26   28   32
   25   17   23   24   27   26   26   25   21   21   28   21   21   20   25   23   20   23   22   19   16   14   22   25   26   27   27   27   27   30   33   26   29
   29   23   19   18   19   26   25   22   26   25   24   23   20   21   22   25   21   19   22   26   27   29   30   28   30   32   34   31   33   31   32   32   39
   22   19   20   26   19   20   15   20   21   24   24   25   24   25   21   26   21   23   25   29   26   25   30   27   26   34   34   29   27   31   32   33   33
   24   20   23   24   21   16   19   19   22   26   20   29   25   25   23   22   17   19   18   20   25   27   27   25   29   32   27   29   31   34   27   28   27
   15   13   17   17   22   18   15   20   25   23   17   24   26   22   26   22   18   19   19   24   24   25   25   24   32   27   35   33   35   32   28   25   24
   17   15   11   19   25   14   10   18   27   25   18   20   24   21   23   20   24   24   17   25   23   28   27   22   32   31   39   43   35   39   38   44   43
   15   19   20   11   18   19   15   17   18   23   17   20   20   24   24   26   21   24   21   28   30   28   32   26   33   33   41   38   47   39   39   35   33
   12   19   13   14   14   19   19   17   18   20   20   18   18   15   13   19   24   20   17   19   18   26   29   30   22   23   31   33   35   35   39   37   34
   12   15   13   10   12   14   14   17   14   12   12   16   18   20   22   25   21   22   19   22   25   32   32   28   23   26   31   33   33   33   26   33   33
   13   12   12   13   14   17   15   15   14   12   16   22   21   21   22   28   26   20   20   19   26   25   28   24   26   30   36   35   33   35   34   26   30
   10    8   14   15   19   11   14   14   14   14   16   17   15   13   23   19   18   23   20   18   27   31   30   31   35   28   31   28   36   34   31   32   30
   10   10    9   13   10    8   14   12   14   11   14   13   12   16   18   16   23   24   27   30   24   17   20   28   29   34   35   26   35   34   32   36   32
   13   13   12   13   10    7   13   16   13   15   14   15   21   19   19   18   16   23   21   25   25   29   26   30   34   33   37   14   33   36   34   34   35
   16   14   14   12    9   11   14   15   14   17   20   18   12   18   20   23   16   20   25   26   32   34   36   38   32   32   30   23   32   29   29   27   21
   11   18   16   11   16   16   16   14   15   17   18   15   19   18   21   22   22   23   28   20   20   25   30   31   21   25   32   29   32   31   37   37   30
   14   12   17   14    9   15   16   11   15   12   14   17   19   16   17   21   17   17   18   17   20   27   33   32   27   27   29   26   27   28   31   25   31
    8   13   16   12   11   14   13   12   16   16   16   19   19   16   17   21   26   23   20   22   25   27   30   32   34   32   31   26   23   29   29   22   32
   13   11   14   15   14   13    7   11   19   16   14   14   15   20   23   30   27   28   25   22   29   27   25   35   29   31   27   32   28   37   37   34   31
   13   18   16   12   16   18   13   25   26   22   20   19   21   20   23   28   27   33   29   27   25   25   27   34   28   31   29   36   25   29   26   25   21
   19   15   12   18   21   20   17   14   15   15   19   20   23   22   19   17   26   30   27   22   23   21   19   20   24   27   18   26   25   23   23   24   22
   11   11   11   21   18   12   15   16   15   18   17   14   18   17   22   18   21   25   20   25   25   25   26   23   19   27   20   26   25   22   22   26   30
   12   15   17   15   14   13   16   17   15   16   16   17   18   22   25   15   17   18   21   19   26   25   27   26   20   18   30   38   28   23   26   29   36
    8    7    6    9   10   14   20   15   12   16   18   23   28   27   27   22   22   22   20   25   23   26   24   23   25   26   22   32   26   25   19   24   29
    6    5   11   12   14   27   21   23   18   22   24   22   22   20   20   20   20   24   15   19   18   25   27   28   28   25   25   22   24   26   23   20   25
   10   10   11    9    9   18   16   20   20   19   21   20   21   19   21   21   20   25   22   26   23   21   20   16   24   25   22   19   22   24   23   25   29
    9   12   17   12   11   13   14   12   18   26   24   20   18   17   14   22   24   17   24   23   24   23   22   14   21   27   26   25   27   26   25   25   27
   14    7   12    8   13   15   13   11   16   19   23   24   21   17   22   20   27   25   25   31   26   30   30   25   29   25   26   23   21   21   31   22   25
```

Figure 11. Binary correlation output for a 32 by 32 window array.

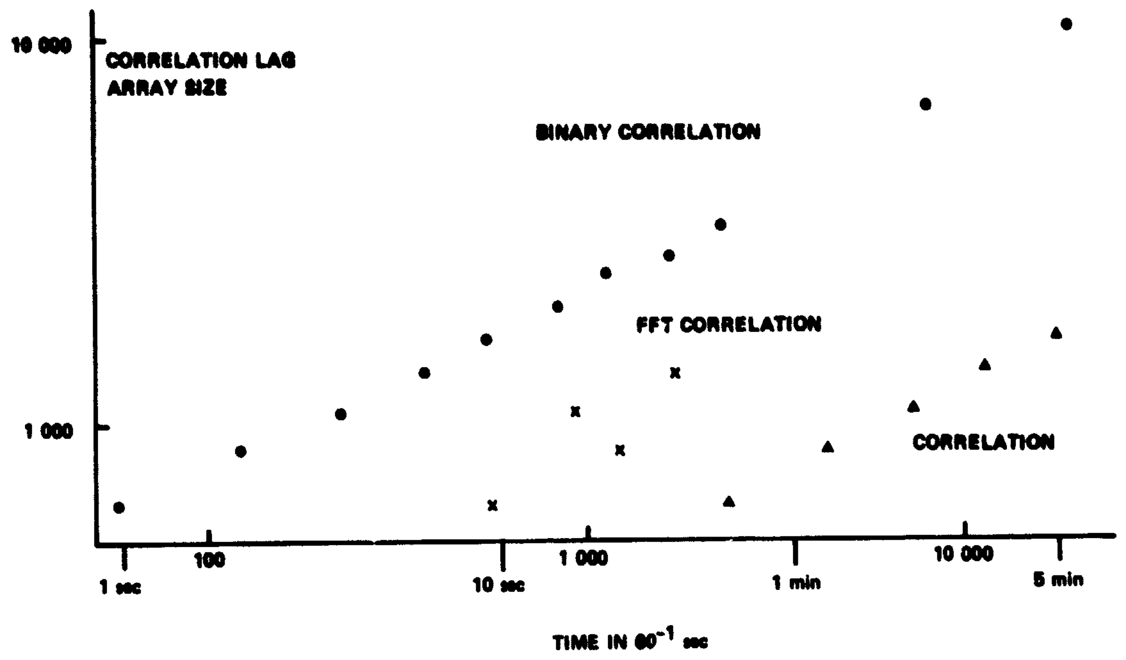

Figure 12. Correlation lag array size versus running time.

TABLE 5. RUNNING TIME VERSUS CORRELATION LAG ARRAY SIZE

Lag Array Size	Running Time in 60^{-1} seconds/Times Slower Than Binary Correlation		
	Binary Correlation	FFT Correlation	Correlation
25 by 25	58	585/10.09	2 450/42.24
29 by 29	124	1267/10.22	4 465/36.01
33 by 33	230	983/4.27	7 532/32.75
37 by 37	381	1778/4.67	11 962/31.4
41 by 41	559		18 097/32.37
45 by 45	850		
49 by 49	1 167		
53 by 53	1 710		
57 by 57	2 377		
81 by 81	8 371		
101 by 101	19 761		

Reference 1 for computing the equivalent number of integer adds for a 32 by 32 window and 64 by 64 picture. The formula presented in Reference 1 is $4(1 + 10\sqrt{M/32})(L - M+1)^2$ equivalent integer adds, where L and M are the picture and window length or width, respectively. For the above mentioned picture and window the number of equivalent integer adds for the SSDA is 47,916 while the number of integer adds for the particular data set shown in Figure 10 is 20,804. Thus, it would appear that the binary correlation could possibly be 2.3 times faster than the SSDA. However, it is also possible that computing equivalent integer adds for various types of programs does not give the complete story, since it is necessary in all the computer programs to execute additional computer statements other than updating the correlation. Otherwise, the binary correlation would be approximately 200 times faster than the FFT correlation. Table 5 indicates that it is not. It is realized that one does not obtain "something for nothing," and the processing time needed to produce a binary boundary map could negate the speed of the binary correlation. However, two points in addition to those mentioned in Section I are worth considering. According to Reference 1 it may become necessary to resort to the use of boundary maps when the channels under consideration are widely separated in spectral wavelength. This is certainly evidenced when one is working with near infrared or thermal imagery which tends to be quite noisy. Secondly, Figure 13 suggests a way to optimize the production of a boundary map.

Figure 13 illustrates the number of variables needed to define various window sizes for the correlation and binary correlation methods using the data set illustrated in Figure 10. Also included in Figure 13 is the number of variables needed to define the location of the boundary elements only in the data set, which is twice the number of boundary points. For the correlation it is necessary to store the entire window array, while for the binary correlation it is necessary only to store 3 X KMAX variables, the start scan, the start column, and the sequence length for each boundary sequence. Thus, Figure 13 indicates the amount of data compression that is possible. Table 6 lists the compression factor for various window sizes, which is the total number of data points divided by 3 X KMAX.

The production of a boundary map could be optimized by recognizing that three variables are needed to define each boundary sequence. Also, from the logic used in calculating IW(K), JW(K), NSQW(K), IP(L), JP(L), and NSQP(L), it is only necessary to determine boundaries within a scan and not necessary to determine boundaries that occur across two adjacent scan lines. Thus, the boundary program described in Section II could be optimized by working with one scan of data at a time instead of two and by eliminating all boundary sequences that contain less than three boundary elements. This would also optimize the compression by assuring that 3 X KMAX is always equal to or less than the number of boundary elements.

It is foreseen that in some cases it may become necessary to subtract mean values from NCOR(I,J) and normalize to produce a proper registration of two data sets. In this respect it is interesting to examine the correlation coefficient in terms of rewards and

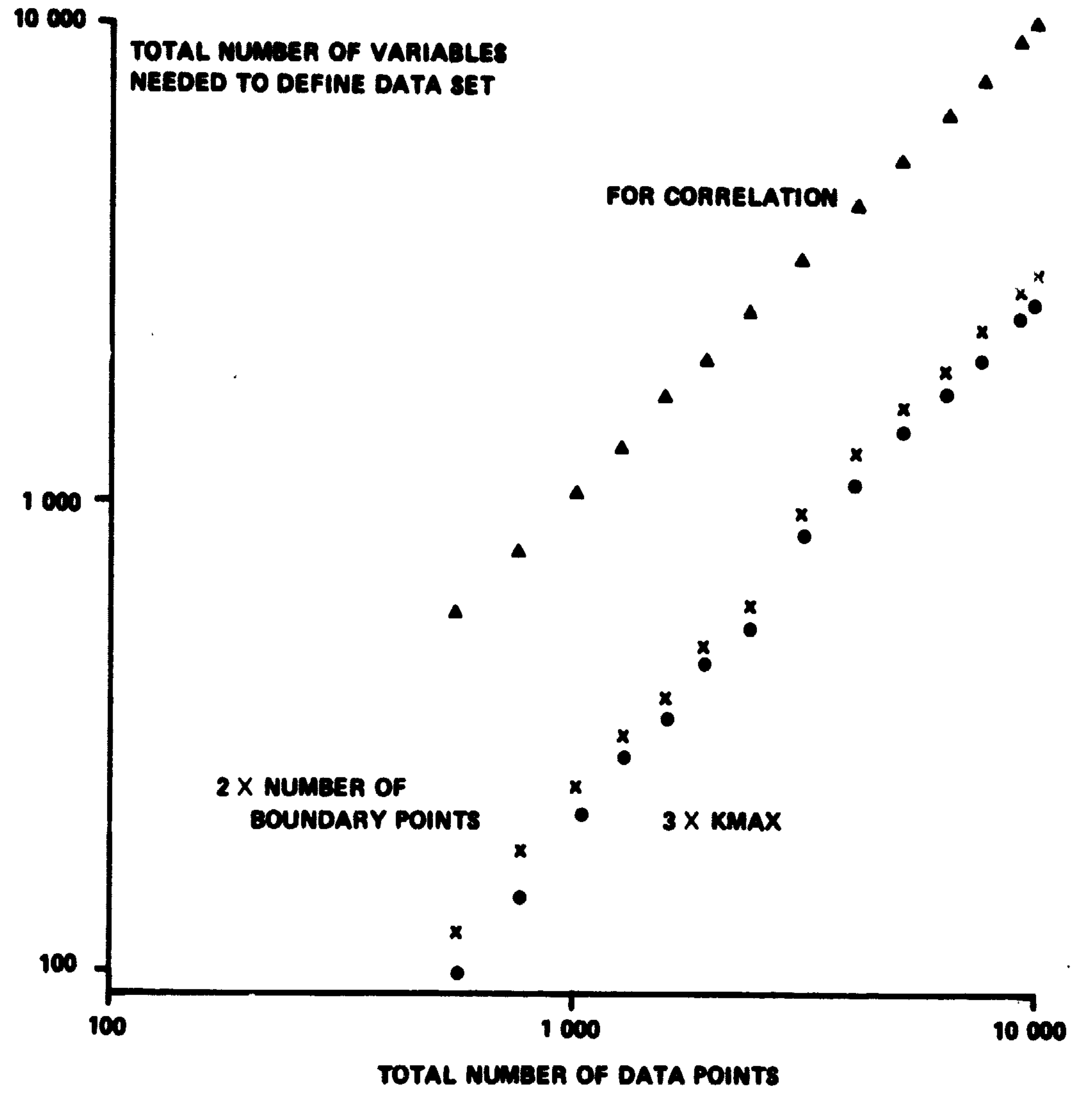

Figure 13. Number of variables needed to define window array.

penalties used in the binary correlation method. The binary correlation method presented in Section III only rewards for boundary elements that match and does not penalize for mismatches of boundary and nonboundary elements. By subtracting mean values and normalizing, it is possible to obtain rewards, for matching boundary elements and matching nonboundary elements, and penalties for mismatches of boundary and nonboundary elements. Before examining the correlation coefficient, it will be necessary to define a few terms.

TABLE 6. COMPRESSION FACTOR FOR WINDOW ARRAY

Window Array Size	Number of Boundary Elements	KMAX	Compression Factor
24 by 24	61	33	5.81
28 by 28	90	48	5.44
32 by 32	122	70	4.88
36 by 36	158	95	4.55
40 by 40	183	115	4.64
44 by 44	237	151	4.27
48 by 48	296	178	4.31
52 by 52	368	229	3.94
56 by 56	450	281	3.72
64 by 64		355	3.85
72 by 72		451	3.83
80 by 80	903	552	3.86
88 by 88		656	3.93
96 by 96		789	3.89
100 by 100	1431	847	3.94

NWB	The number of boundary elements in the window.
NPB(I,J)	The number of boundary elements in the picture that corresponds to the area covered by the window. NPB(I,J) changes as the window is moved to different I and J locations.

W	The mean value of the data in the window, W=NWB/[(NROW)(NCOL)].
P(I,J)	The mean value of the data in the picture that corresponds to the area covered by the window, P(I,J)=NPB(I,J)/[(NROW)(NCOL)].

The numerator of the correlation coefficient, C(I,J) can then be written as

$$C(I,J) = K_1 \cdot NCOR(I,J) + K_2 \cdot [NPB(I,J) - NCOR(I,J)]$$
$$+ K_3 \cdot [NWB - NCOR(I,J)] + K_4 \cdot [NROW \cdot NCOL - NPB(I,J)$$
$$+ NCOR(I,J) - NWB] \quad , \qquad (.1)$$

where

$$K_1 = [1 - W][1 - P(I,J)] \;,\; K_2 = -W[1 - P(I,J)],\; K_3 = -[1 - W]P(I,J)$$

and

$$K_4 = W \cdot P(I,J)$$

Since $0 < W$, $P(I,J) < 1$, then $K_1, K_4 > 0$ and $K_2, K_3 < 0$, and K_1 and K_4 are always greater than K_2 and K_3 in absolute ma⁻nitude. If the constants, K, are thought of in terms of rewardε and penalties, then the first term of equation (11) is a reward times the number of boundary elements that match in the window and picture. The second and third terms, since they are negative, penalize the mismatch of boundary and nonboundary elements in the picture and window, respectively. The last term is a reward for nonboundary elements that match in the picture and window. It is interesting to note that for $W+P(I,J) < 1$, then $K_1 > K_4 > K_2, K_3$; that for $W+P(I,J)=1$, then $K_1 = K_4 > K_2, K_3$; and that for $W+P(I,J) > 1$, then $K_4 > K_1 > K_2, K_3$. This indicates that the binary correlation procedure gives a larger reward for matching up whatever occurs the least in the window or picture, boundaries or nonboundaries. This is also what a human observer would tend to do. To determine a normalization factor for equation (11), consider a perfect match for the window and picture; i.e., NWB=NPB(I,J)=NCOR(I,J) and $W=P(I,J) < 1$. Equation (11) becomes

$$C(I,J) = (1 - W)^2 \cdot NWB + W^2 \cdot (NROW \cdot NCOL - NWB)$$

$$= [1 - P(I,J)]^2 \cdot NPB(I,J) + [P(I,J)]^2$$

$$[NROW \cdot NCOL - NPB(I,J)] \qquad (12)$$

Thus, for a nonperfect match, the normalization factor should be some combination of the two expressions in equation (12). Since the two expressions in equation (12) are the variances of the window and picture respectively, the correct normalization factor would be the square root of the products of the two expressions to produce a correlation coefficient.

It does not appear that computing the correlation coefficient would add a significant amount of running time to the binary correlation routine. This is because the mean value and variance of the window need only be calculated once and $NWB=KMAX+\sum_{K=1}^{KMAX} NSQW(K)$. The calculation that would require additional time and storage is determining NPB(I,J) for the mean value and variance of the picture for each shift of the window. In most cases, however, it is anticipated that it will be necessary to compute only NCOR(I,J).

George C. Marshall Space Flight Center
National Aeronautics and Space Administration
Marshall Space Flight Center, Alabama, November 19, 1973

APPENDIX A. BINARY CORRELATION LISTING

```
5245, CAMPBELL      ,140145,30,               FORTRAN SOURCE LIST
      ISN          SOURCE STATEMENT

        0 $IBFTC MAIN     DECK
        1          DIMENSION NW(200),FMT(12),NCOR(101,101),IW(1324),JW(1324)
        2          DIMENSION NSQW(1324),IP(3300),JP(3300),NSQP(3300)
        3          READ 300,NROW,NCOL
        6      300 FORMAT(2I5)
        7          PRINT 302,NROW,NCOL
       10      302 FORMAT(1X,5HNROW=,I4,1X,5HNCOL=,I4,/)
       11          READ(5,301)(FMT(L),L=1,12)
       16      301 FORMAT(12A6)
       17          NROW1=NROW+1
       20          NCOL1=NCOL+1
       21          NCOR(NROW1,NCOL1) =0
       22          K=0
       23          DO 20 I=1,NROW
       24          IN=-1
       25          READ(5,FMT)(NW(I1),I1=1,NCOL)
       32          DO 10 J=1,NCOL
       33          NCOR(I,J)=0
       34          IF(NW(J).EQ.0)GO TO 5
       37          IF(IN.GT.0) GO TO 3
       42          K=K+1
       43          IW(K)=I-1
       44          JW(K)=J-1
       45          NSQW(K)=0
       46          IN=1
       47          GO TO 10
       50        3 NSQW(K)=NSQW(K)+1
       51          GO TO 10
       52        5 IN=-1
       53       10 CONTINUE
       55       20 CONTINUE
       57          KMAX=K
       60          READ(5,301)(FMT(L),L=1,12)
       65          NROW2=NROW+NROW
       66          NCOL2=NCOL+NCOL
       67          L=0
       70          DO 40 I=1,NROW2
       71          IN=-1
       72          READ(5,FMT)(NW(I1),I1=1,NCOL2)
       77          DO 30 J=1,NCOL2
      100          IF(NW(J).EQ.0)GO TO 25
      103          IF(IN.GT.0) GO TO 23
      106          L=L+1
      107          NSQP(L)=0
      110          IP(L)=I
      111          JP(L)=J
      112          IN=1
      113          IF(I.GT.NROW1)GO TO 30
      116          LMAX1=L
      117          GO TO 30
      120       23 NSQP(L)=NSQP(L)+1
      121          GO TO 30
      122       25 IN=-1
      123       30 CONTINUE
      125       40 CONTINUE
```

```
5245, CAMPBELL      ,140145,30,              FORTRAN SOURCE LIST MAIN
      ISN          SOURCE STATEMENT

      127          LMAX=L
      130          KSTRT=1
      131          ISUM=0
      132          CALL TIMNOW(IT1)
      133          DO 100 L=1,LMAX1
      134          NU=JP(L)+NSQP(L)
      135          DO 80 K=1,KMAX
      136          IF(IW(K).GE.IP(L))GO TO 100
      141          I=IP(L)-IW(K)
      142          IF(I.NE.1)GO TO 45
      145          IF(NU.GT.JW(K))GO TO 46
      150          GO TO 100
      151       45 IF(NU.LE.JW(K))GO TO 80
      154       46 JL=JP(L)-JW(K)-NSQW(K)
      155          IF(JL.GT.NCOL1)GO TO 80
      160          JU=NU-JW(K)
      161          IF(JL.LT.1)GO TO 70
      164          IF(NSQP(L).LE.NSQW(K))GO TO 53
      167          MINL=NSQW(K)
      170          GO TO 54
      171       53 MINL=NSQP(L)
      172       54 IF(MINL.GT.0)GO TO 60
      175          IF(JU.LE.NCOL1)GO TO 55
      200          JU=NCOL1
      201       55 DO 56 J=JL,JU
      202          ISUM=ISUM+1
      203       56 NCOR(I,J)=NCOR(I,J)+1
      205          GO TO 80
      206       60 JM=JU-MINL
      207          NAD=0
      210          IF(JU.LE.NCOL1)GO TO 64
      213          JU=NCOL1
      214       64 DO 68 J=JL,JU
      215          ISUM=ISUM+1
      216          IF(J.GT.JM)GO TO 66
      221          IF(NAD.GT.MINL)GO TO 68
      224          NAD=NAD+1
      225          GO TO 68
      226       66 NAD=NAD-1
      227       68 NCOR(I,J)=NCOR(I,J)+NAD
      231          GO TO 80
      232       70 NSPAN=JW(K)+NSQW(K)+1
      233          IF(JP(L).GT.JW(K))GO TO 71
      236          IF(NSPAN.GE.NU)GO TO 69
      241          NAD=NSQW(K)+1
      242          GO TO 75
      243       69 NAD=JU
      244          GO TO 75
      245       71 IF(NSPAN.GE.NU)GO TO 73
      250          NAD=NSPAN-JP(L)
      251          IF(NSQP(L).GT.NSQW(K))GO TO 74
      254          MINL=NSQP(L)
      255          GO TO 76
      256       74 MINL=NSQW(K)
      257       76 JM=JU-MINL
```

```
5245, CAMPBELL      ,140145,00,                FORTRAN SOURCE LIST MAIN
             ISN         SOURCE STATEMENT

             260         JL=1
             261         GO TO 64
             262      73 NAD=NSQP(L)+1
             263      75 JM=JU-NAD+2
             264         DO 78 J=1,JU
             265         ISUM=ISUM+1
             266         IF(J.LT.JM)GO TO 78
             271         NAD=NAD-1
             272      78 NCOR(I,J)=NCOR(I,J)+NAD
             274      80 CONTINUE
             276     100 CONTINUE
             300         LMAX1=LMAX1+1
             301         DO 200 L=LMAX1,LMAX
             302         NU=JP(L)+NSQP(L)
             303      52 DO 180 K=KSTRT,KMAX
             304         I=IP(L)-IW(K)
             305         IF(I.GT.NROW1)GO TO 85
             310         IF(NU.LE.JW(K))GO TO 180
             313         JL=JP(L)-JW(K)-NSQW(K)
             314         IF(JL.GT.NCOL1)GO TO 180
             317         JU=NU-JW(K)
             320         IF(JL.LT.1)GO TO 701
             323         IF(NSQP(L).LE.NSQW(K))GO TO 531
             326         MINL=NSQW(K)
             327         GO TO 541
             330     531 MINL=NSQP(L)
             331     541 IF(MINL.GT.0)GO TO 601
             334         IF(JU.LE.NCOL1)GO TO 552
             337         JU=NCOL1
             340     552 DO 561 J=JL,JU
             341         ISUM=ISUM+1
             342     561 NCOR(I,J)=NCOR(I,J)+1
             344         GO TO 180
             345     601 JM=JU-MINL
             346         NAD=0
             347         IF(JU.LE.NCOL1)GO TO 641
             352         JU=NCOL1
             353     641 DO 681 J=JL,JU
             354         ISUM=ISUM+1
             355         IF(J.GT.JM)GO TO 661
             360         IF(NAD.GT.MINL)GO TO 681
             363         NAD=NAD+1
             364         GO TO 681
             365     661 NAD=NAD-1
             366     681 NCOR(I,J)=NCOR(I,J)+NAD
             370         GO TO 180
             371     701 NSPAN=JW(K)+NSQW(K)+1
             372         IF(JP(L).GT.JW(K))GO TO 711
             375         IF(NSPAN.GE.NU)GO TO 691
             400         NAD=NSQW(K)+1
             401         GO TO 751
             402     691 NAD=JU
             403         GO TO 751
             404     711 IF(NSPAN.GE.NU)GO TO 731
             407         NAD=NSPAN-JP(L)
```

```
5245, CAMPBELL      ,140145,00,              FORTRAN SOURCE LIST MAIN
          ISN        SOURCE STATEMENT

          410        IF(NSQP(L).GT.NSQW(K))GO TO 741
          413        MINL=NSQP(L)
          414        GO TO 761
          415    741 MINL=NSQW(K)
          416    761 JM=JU-MINL
          417        JL=1
          420        GO TO 641
          421    731 NAD=NSQP(L)+1
          422    751 JM=JU-NAD+2
          423        DO 781 J=1,JU
          424        ISUM=ISUM+1
          425        IF(J.LT.JM)GO TO 781
          430        NAD=NAD-1
          431    781 NCOR(I,J)=NCOR(I,J)+NAD
          433    180 CONTINUE
          435        GO TO 200
          436     85 KSTRT=K+1
          437        GO TO 52
          440    200 CONTINUE
          442        CALL TIMNOW(IT2)
          443        PRINT 5555,ISUM
          444   5555 FORMAT(5X,7HISUM = ,I7)
          445        JTIME=IT2-IT1
          446        PRINT 318
          447    318 FORMAT(5X,50H**************************************************)
          450        PRINT 317,JTIME
          451    317 FORMAT(/////,5X,15HELAPSED TIME = ,I5,11H/60 SECONDS,//////)
          452        PRINT 1100,KMAX,LMAX
          453   1100 FORMAT(5X,7HKMAX = ,I5,5X,7HLMAX = ,I5)
          454        PRINT 318
          455        PRINT 999
          456    999 FORMAT(5X,15HREGIS4 COMPLETE)
          457        JSTRT=1
          460        JSTOP=20
          461    405 PRINT 403
          462    403 FORMAT(1H1,5X,18HCORRELATION MATRIX)
          463        DO 401 I1=1,NROW1
          464        PRINT 402,(NCOR(I1,J),J=JSTRT,JSTOP)
          471    402 FORMAT(1X,20(I4,1X))
          472    401 CONTINUE
          474        IF(JSTOP.EQ.NCOL1)GO TO 406
          477        JSTRT=JSTOP+1
          500        JSTOP=JSTOP+20
          501        IF(JSTOP.LE.NCOL1)GO TO 405
          504        JSTOP=NCOL1
          505        GO TO 405
          506    406 CONTINUE
          507        STOP
          510        END
```

APPENDIX B. CORRELATION ROUTINE LISTING

```
5245, CAMPBELL      ,140145,00,              FORTRAN SOURCE LIST
        ISN        SOURCE STATEMENT

          0 $IBFTC COR      DECK
          1        SUBROUTINE COR(NP,NW,NCOR)
            C*************************************************************************
            C   THIS SUBROUTINE CONTAINS THE STANDARD CORRELATION CALCULATION        *
            C*************************************************************************
          2        DIMENSION NW(56,56),NP(112,112),NCOR(57,57),FMT(12)
          3        COMMON/NAME1/ NCOL
          4        COMMON/NAME2/ NROW
          5        COMMON/NAME4/ FMT
          6        READ (5,301) (FMT(L),L=1,12)
         13   301  FORMAT(12A6)
         14        DO 10 I1=1,NROW
         15        READ(5,FMT) (NW(I1,K1), K1=1,NCOL)
         22    10  CONTINUE
         24        NROW=NROW+NROW
         25        NCOL=NCOL+NCOL
         26        READ (5,301) (FMT(L),L=1,12)
         33        DO 20 I2=1,NROW
         34        READ(5,FMT) (NP(I2,K2), K2=1,NCOL)
         41    20  CONTINUE
         43        NROW=NROW/2
         44        NCOL=NCOL/2
         45        CALL TIMNOW(IT1)
         46        NROW1=NROW+1
         47        NCOL1=NCOL+1
         50        DO 1 I=1,NROW1
         51        DO 2 J=1,NCOL1
         52        NCOR(I,J)=0
         53        DO 3 K=1,NROW
         54        I1=I+K-1
         55        DO 4 L=1,NCOL
         56        J1=J+L-1
         57        NCOR(I,J)=NCOR(I,J)+NW(K,L)*NP(I1,J1)
         60     4  CONTINUE
         62     3  CONTINUE
         64     2  CONTINUE
         66     1  CONTINUE
         70        CALL TIMNOW(IT2)
         71        JTIME=IT2-IT1
         72        PRINT 318
         73   318  FORMAT(5X,50H**************************************************)
         74        PRINT 317,JTIME
         75   317  FORMAT(/////,5X,15HELAPSED TIME = ,I5,11H/60 SECONDS,//////)
         76        PRINT 318
         77        PRINT 999
        100   999  FORMAT(5X,15HCOR    COMPLETE)
        101        RETURN
        102        END
```

APPENDIX C. FAST FOURIER TRANSFORM CORRELATION LISTING

```
5245, CAMPBELL      ,140145,00,                    FORTRAN SOURCE LIST
      ISN         SOURCE STATEMENT

         0 $IBFTC MAIN
         1        DIMENSION XNWR(64,64),XNWI(64,64),XNPR(64,64),XNPI(54,64)
         2        DIMENSION IA(64,64)
         3        READ 1,NROW,NCOL
         6      1 FORMAT(2I5)
         7        CALL IN(XNPR,XNPI,NROW,NCOL)
        10        CALL IN(XNWR,XNWI,NROW,NCOL)
        11        NTOT=NROW*NCOL
        12        CALL TIMNOW(IT1)
        13        CALL FFT7(XNPR,XNPI,NTOT,NROW,NCOL,1)
        14        CALL FFT7(XNPR,XNPI,NTOT,NROW,NTOT,1)
        15        CALL FFT7(XNWR,XNWI,NTOT,NROW,NCOL,1)
        16        CALL FFT7(XNWR,XNWI,NTOT,NROW,NTOT,1)
        17        XNTOT=NTOT
        20        DO 7 I=1,NROW
        21        DO 8 J=1,NCOL
        22        STORE=XNPR(I,J)*XNWR(I,J)+XNPI(I,J)*XNWI(I,J)
        23        XNPI(I,J)=(XNPI(I,J)*XNWR(I,J)-XNWI(I,J)*XNPR(I,J))/XNTOT
        24        XNPR(I,J)=STORE/XNTOT
        25      8 CONTINUE
        27      7 CONTINUE
        31        CALL FFT7(XNPR,XNPI,NTOT,NROW,NCOL,-1)
        32        CALL FFT7(XNPR,XNPI,NTOT,NROW,NTOT,-1)
        33        CALL TIMNOW(IT2)
        34        ITIME=IT2-IT1
        35        PRINT 14
        36        PRINT 13,ITIME
        37     13 FORMAT(1H1,5X,8HITIME = ,I5,3H/60)
        40        PRINT 14
        41     14 FORMAT(60H********************************************************
                1****)
        42        CALL OUT(XNPR,NROW,NCOL)
        43        STOP
        44        END
```

REFERENCES

1. Barnea, D.I.; and Silverman, J.F.: A Class of Algorithms for Fast Digital Image Registration. IEEE Transaction on Computers, vol. C-21, no. 2, February 1972, pp. 179-186.

2. Anuta, P.E.: Spatial Registration of Multispectral and Multitemporal Digital Imagery Using Fast Fourier Transform Techniques. IEEE Transactions of Geoscience Electronics, vol. GE-8, no. 4, October 1970, pp. 353-368.

3. Anuta, P.E.: Digital Registration of Multispectral Video Imagery. S.P.I.E. Journal, vol. 7, September 1969, pp. 168-175.

4. Maynard, H.W.: An Evaluation of Ten Fast Fourier Transform (FFT) Programs. Technical Report ECOM-5476, U.S. Army Electronics Command, Fort Monmouth, New Jersey, March 1973.

NASA-Langley, 1974